Mathématiques et sciences humaines
Mathematics and social sciences

199 | 2012
Psychologie et mathématiques

Compression mechanisms in working memory

Mécanismes de compression en mémoire de travail

Benoît Lemaire, Vivien Robinet and Sophie Portrat

Electronic version
URL: http://journals.openedition.org/msh/12301
DOI: 10.4000/msh.12301
ISSN: 1950-6821

Publisher
Centre d'analyse et de mathématique sociales de l'EHESS

Printed version
Date of publication: 15 September 2012
Number of pages: 71-84
ISSN: 0987-6936

Electronic reference
Benoît Lemaire, Vivien Robinet and Sophie Portrat, « Compression mechanisms in working memory », *Mathématiques et sciences humaines* [Online], 199 | 2012, Online since 04 December 2012, connection on 02 May 2019. URL : http://journals.openedition.org/msh/12301 ; DOI : 10.4000/msh.12301

Math. Sci. hum / Mathematics and Social Sciences (50e année, n° 199, *2012*(3), p. 71-84)

COMPRESSION MECHANISMS IN WORKING MEMORY

Benoît LEMAIRE[1], Vivien ROBINET[2], Sophie PORTRAT[1]

RÉSUMÉ – Mécanismes de compression en mémoire de travail

La capacité de la mémoire de travail est limitée et de nombreux travaux cherchent depuis plusieurs décennies à estimer sa valeur. Or, dès lors que l'on considère des stimuli contenant des régularités, des mécanismes de compression d'information opèrent probablement, changeant ainsi le point de vue sur cette capacité. À travers une étude comportementale suivie d'une simulation computationnelle, nous cherchons à montrer que la capacité de la mémoire de travail ne se résume pas à un nombre fixe d'items. Nous présentons tout d'abord un cadre théorique dans le champ de la théorie de l'information pour analyser ce point de vue. Ensuite, nous décrivons les résultats d'une expérience visant à étudier les effets de certaines régularités dans les stimuli sur les performances de rappel en mémoire, ainsi qu'une simulation utilisant un modèle de « chunking » et deux modèles de mémoire différents. Nos résultats montrent qu'il devient probablement erroné de considérer la capacité de la mémoire de travail comme un nombre fixe d'items et qu'il est préférable de l'exprimer en terme de quantité d'information.

MOTS-CLÉS – Compression d'informations, Expérience, Longueur de description minimale, Mémoire de travail, Simulations, Théorie de l'information

ABSTRACT – *Working memory capacity is limited and much work has been done for decades on estimating its value. However, if stimuli contain redundancies, compression mechanisms probably appear which change the point of view on that capacity. By means of a behavioral study and a computational simulation, we aim at showing that working memory is not just a fixed number of items. We first present a theoretical framework in the domain of information theory in order to analyse this point of view. Then, we show the results of an experiment studying the effects of some regularities on memory recall performance, as well as a simulation using a model of chunking and two different memory models. Our results show that it is probably wrong to consider working memory capacity as a fixed number of items. It is better to express it in terms of a quantity of information.*

KEYWORDS – Information compression, Information theory, Human experiment, Length minimum description, Simulation, Working memory

1. INTRODUCTION

Assessing the capacity of human working memory (WM) has been a constant concern for researchers since the seminal paper from Miller [1956] which estimated that capacity to be about 7±2 items. The capacity is often measured by presenting participants with sequence of letters, numbers, visual items, etc. and asking them to immediately recall as much items as possible. If participants have some prior knowledge about relationships between items, they can improve their performance by

[1] Laboratoire de psychologie et neuro-cognition, Université Grenoble 2, BP 47, 38040 Grenoble Cedex 9, Benoit.Lemaire@upmf-grenoble.fr, Sophie.Portrat@upmf-grenoble.fr

[2] UMR Espace-Dev, Centre IRD de Cayenne, Route de Montabo, BP 165, 97323 Cayenne Cedex (Guyane), Vivien.Robinet@guyane.univ-ag.fr

grouping several items into what is called a *chunk*. It is defined as "*a collection of elements having strong associations with one another, but weak associations with elements within other chunks*" [Gobet *et al*., 2001]. For instance, although it contains 7 items, memorizing U S S R U S A is very easy because it is composed of only two chunks (USSR and USA).

Classically, researchers make sure to carefully control these relationships and therefore avoid any extra regularities in the input. As a consequence, the dominant trend in the scientific community is that all chunks take the same space in WM. However, in everyday life, inputs are full of redundancies that we may take advantage of to encode and memorize more information. Indeed, information theory tells us that redundancies in the input lead to code information in a shorter way, because redundancies allow compression. Compression mechanisms may therefore occur in WM and some very recent papers address that issue. For example, Mathy & Feldman [2012] study how humans take into account regularities in sequences of numbers in order to store them in a efficient way in memory.

This kind of compression could be called a within-stimulus compression. All stimuli are independent but each one may contain some redundancy that allows compression. Another kind of compression is between-stimuli. Compression is made possible because there are some redundancies over all stimuli. This last approach is interesting because there is no need to rely on participant's prior knowledge since redundancies can be built over trials. Brady *et al*. [2009] presented an experiment in which participants were asked to recall colors in a display in which some pairs tend to occur next to each other more frequently than other pairs. They showed that participants take this redundancy into account to compress the representation, store more information in memory and therefore improve their performance.

Compression may be seen as a byproduct of the stimulus representation, which is based on previous knowledge about redundancy that may occur in the input. Representing (and thus memorizing) USSRUSA as two chunks, requires both USSR and USA to be recognized as a chunk, i.e. as a sufficiently strong regularity. As an example, in the field of verbal stimuli, compression is closely related to the classic speech segmentation task [Perruchet & Vinter, 1998; Robinet & Lemaire, 2009], which has been broadly studied in the literature [Saffran *et al*., 1996; Swingley, 2005].

Compression mechanisms may occur because they offer a way to store more information in the limited WM buffer. There is no reason to deny the fact that WM may included these powerful mechanisms. If they really exist, then WM capacity may be better expressed as a fixed quantity of information, instead of as a fixed amount of chunks which is the unit that is often used since Miller's work.

2. COMPRESSION

Our hypothesis is that memory encoding would be based on an economy principle that would tend to store the most concise (or simplest [Chater & Vitanyi, 2003]) structures. This can be modeled by considering the amount of information that each stimulus contains.

2.1 ALGORITHMIC COMPLEXITY

A rigorous and general formalism called Algorithmic complexity (or Kolmogorov complexity) [Solomonoff, 1964; Chaitin, 1966; Kolmogorov, 1968] has been defined to

quantify the amount of information carried by a finite discrete stimulus *s*. It is defined as the "length of the shortest program *p*, that is able to reproduce the stimulus *s*, and then halt". The program *p* is run on a Turing machine *Ti* [Turing, 1936]. It is an automaton whose behavior is defined by an action table *i*.

Within this theoretical framework, it has been shown that the complexity $K_{Ti}(s)$, or amount of information carried by *s*, is independent on the particular Turing machine *Ti* used to run the program. This invariance theorem [Solomonoff, 1964; Chaitin, 1966; Kolmogorov, 1968]:

$$\exists \lambda(T1,T2),\ \forall s,\ K_{T2}(s) \leq K_{T1}(s) + \lambda(T1,T2)$$

ensures the generality of the measure. It is true up to an additive constant λ only depending on the action tables of the considered Turing machines *T1* and *T2*. In particular, it is independent on *s*. The constant $\lambda(T1,T2)$ is the size of the emulator able to translate the instructions of the Turing machine *T1* into instructions for *T2*.

Without loss of generality, it is possible to consider an universal Turing machine *U* whose programs p^U are a concatenation of the definition *i* of the action table characterizing the particular Turing machine *Ti*, followed by a program p^i written in language *Ti:*

$$p^U = i \bullet p^i$$

where $\bullet$ is the concatenation operator. One of the most important theorem in algorithmic complexity is probably the Coding Theorem [Levin, 1974] establishing the relation between the size $K_U(s)$ of the shortest program p^U for *s*, and the probability $Q_U(s)$ that a random self-delimiting program produces *s* as an output:

$$K_U(s) = \log_2(1 / Q_U(s)) + O(1) \qquad (1)$$

The algorithmic complexity of a given stimulus *s* is intractable in practice, nonetheless, the latter theorem gives a hint on the way complexity can be deduced from probability. One way to model these compression mechanisms is therefore to rely on information theory [Shannon, 1948]. Within this formalism, the quantity of information carried by a stimulus *s* is estimated by its codelength *C(s)*:

$$C(s) = \log_2(1 / P(s)) \qquad (2)$$

This is the quantity of information (expressed in bits) necessary to isolate the stimulus *s* from all other possible stimuli *s'*. The difference between eq. (1) and eq. (2) is that Q_U is an absolute probability measure on the stimuli space induced by the natural Solomonoff's [1960] probability measure on the program space, whereas *P* is simply the empirical probability of the stimulus in a set of observations.

Rare events carry more information than frequent ones. For example, in the following sequence of independent and identically distributed stimuli

$$a, b, a, a, c, d, a, b$$

the stimulus *a* is more frequent than *d* ($P(a) = 1/2$, $P(d) = 1/8$). Trying to predict the next stimulus in the sequence, leads *a* to be the best candidate with $P(a) = 1/2$. Thus, knowing that *a* is the next stimulus gives only $C(a) = \log_2(2) = 1bit$ of information: the bit is the quantity of information necessary to separate two eventualities having equal probabilities. In the present case, it is the quantity of information necessary to separate the stimulus *a* from the other stimuli *b*, *c* and *d*.

A sequence of independent and equally distributed stimuli can be optimally compressed using a Shannon-Fano code associating a codeword of length *C(s)*, to each

stimulus *s*. In this particular case, *C(s)* cannot differ significantly from $K_U(s)$ [Leung-Yan-Cheong & Cover, 1978].

2.2 MINIMUM DESCRIPTION LENGTH

Compressing information addresses the problem of separating compressible information from noise (incompressible information). In the formalism of algorithmic complexity, this bias-variance dilemma may be represented in the following manner. A program *p* for *s* may be separated into a compressible part p_Σ and an incompressible part $p_{s|\Sigma}$ (noise):

$$length(p) = length(\, p_\Sigma\,) + length(\, p_{s|\Sigma}\,) + O(1)$$

This separation between model and noise may be formalized using the Kolmogorov structure functions [Kolmogorov, 1974]. These functions are broadly studied in Vereshchagin & Vitanyi [2002]. The structure of the following paragraphs is based on this publication.

The stimulus *s* is discrete and composed of symbols from a finite alphabet, so without loss of generality, *s* can be represented by a finite length binary string:

$s \in \Sigma \subseteq \{0,1\}^*$ where $\{0,1\}^*$ is the set of all finite length binary sequences.

The compressible part (or "model") is equivalently defined by its program p_Σ or by the set of its elements $\Sigma = \{s_1,\ldots,s_m\}$. Indeed, p_Σ can be seen as the shortest program able to separate $\{s_1,\ldots,s_m\}$ from all other finite length binary strings $\{0,1\}^*$.

We consider the following quantities:

- $K(\Sigma)$ is the size of the shortest program p_Σ able to isolate the set $\{s_1,\ldots,s_m\}$ from all other possible stimuli.
- $K(s|\Sigma)$ is the size of the shortest program $p_{s|\Sigma}$ able to isolate $s = s_i \in \Sigma$ from all other possible $s_j \in \Sigma$.
- $\log_2(\mathrm{Card}(\Sigma))$ is the minimal number of bits necessary to isolate one index $i \in \{1,\ldots,m\}$ from the others.

Because $\log_2(\mathrm{Card}(\Sigma))$ is the highest quantity of information necessary to isolate s_i without using its structure :

$$K(s|\Sigma) \le \log_2(\mathrm{Card}(\Sigma)) + 0(1)$$

and the difference

$$\delta(s|\Sigma) = \log_2(\mathrm{Card}(\Sigma)) - K(s|\Sigma)$$

is called *randomness deficiency*.

The three Kolmogorov structure functions are:

- The minimal randomness deficiency estimator:

$$\beta_s(\alpha) = \min_\Sigma\{\delta(s|\Sigma) : s \in \Sigma,\ K(\Sigma) \le \alpha\}$$

- The maximum likelihood estimator:

$$h_s(\alpha) = \min_\Sigma\{\log_2(\mathrm{Card}(\Sigma)) : s \in \Sigma,\ K(\Sigma) \le \alpha\}$$

- The minimum description length estimator:

$$\lambda_s(\alpha) = \min_{\Sigma}\{K(\Sigma) + \log_2(\mathrm{Card}(\Sigma)) : s \in \Sigma,\ K(\Sigma) \leq \alpha\}$$

When increasing the threshold α, the model Σ tends to catch more and more information from the stimulus s. Increasing model complexity may lead to overtraining (the model catches some incompressible information carried by s) and overfitting (the model catches information not carried by s). Both increase the generalization error and are difficult to distinguish in practical cases.

When increasing α :

- The minimal randomness deficiency estimator minimizes $\delta(s|\Sigma)$ and thus selects the smallest typical set Σ for s. When Σ contains all the compressible information, the statistic remains constant.
- The maximum likelihood estimator minimizes $\mathrm{Card}(\Sigma)$. It catches the most compressible information first, and then incompressible information, thus leading to an overtraining of the data s.
- The minimum description length estimator catches the most compressible information first and then remains constant over incompressible information: adding one bit to $K(\Sigma)$ decreases $\log_2(\mathrm{Card}(\Sigma))$ by one bit.

Vitanyi [2005] provides detailed information about the properties of the Kolmogorov structure functions. The behavior of the three estimators are summarized in Figure 1.

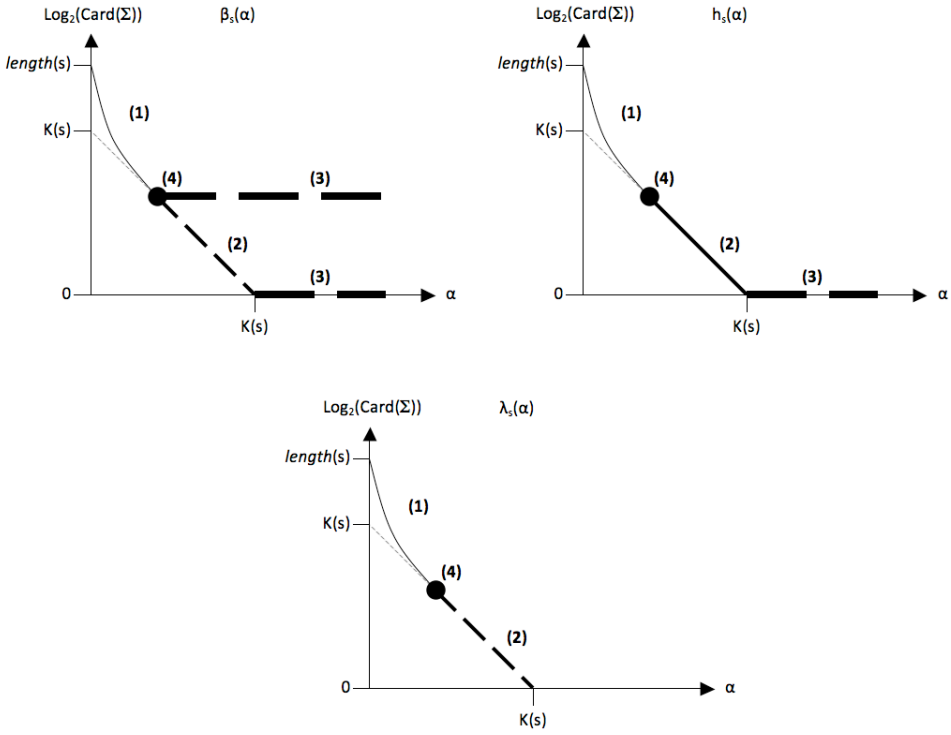

FIGURE 1. Evolution of the model Σ with respect to overtraining and overfitting, when increasing its complexity α.
(1) thin line corresponds to undertraining, (2) normal line to overtraining and (3) thick line to overfitting. (4) the dot represent the minimal sufficient statistic. Dashed line corresponds to a constant value of the statistic

While it remains compressible information, adding 1 bit to the model Σ decreases $\log_2(\text{Card}(\Sigma))$ of at least 1 bit, thus part (1) is strictly convex. It becomes linear in part (2) when the model catches incompressible information. Catching extra-information does not change $\log_2(\text{Card}(\Sigma))$.

The maximum likelihood estimator tends to overtrain the model. β_s and λ_s do not suffer from this drawback (see Figure 2).

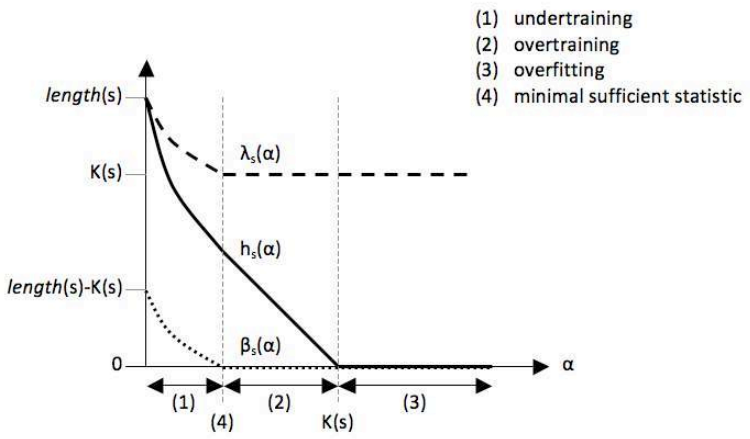

FIGURE 2. Evolution of the three Kolmogorov statistics as functions of α

The minimal randomness deficiency estimator is not computable in practice with a sufficient precision, thus we used the minimum description length (MDL) to estimate the model complexity. For this work, we focused on the simplest MDL estimator called MDL two-part coding [Rissanen, 1978], in which the two parts, "model" and "noise", are explicitly separated, such as p_Σ and $p_{s|\Sigma}$ in the previous formalism. The "model" part catches regularities that are sufficiently strong to be used in order to compress the stimulus. The "noise" part contains non-significant information that are nonetheless necessary to reconstruct the stimuli.

The codelength of each part is computed using a Shannon-Fano code [Shannon, 1948]. The criterion used to select the best model Σ for the set of stimuli is the overall codelength of the two parts. Using this formalism allows to solve the bias-variance dilemma without explicitly specifying a threshold in order to separate regularity and noise.

We now present both an experiment and a simulation using a chunking model which is based on the ideas presented so far.

3. EXPERIMENT

Because we suspect that frequency plays a role in the way information is encoded into memory, in this experiment we aimed at experimentally test these ideas by varying the frequency of chunks exposure and observe if the capacity of WM is affected or not.

Such an experiment was not easy to set up. Because we aimed at properly control the frequency of the chunks exposure, participants should not have known the chunks beforehand and individual differences in memory strategies should have been as much

as possible minimized. For these reasons, we chose to set up an experiment involving visuo-spatial WM. Indeed, while two words or letters are most probably differently linked together across individual semantic networks, there is no reason for two locations on a computer screen to be more strongly associated for one participant than for another. Hence, in our experiment, first, the chunks should not have been previously encountered per se and, second, the stimuli should not be rehearsed mentally using the phonological loop [Baddeley, 1986]. This visuo-spatial WM task, that will be called *location task* from that point, is inspired by the Corsi-blocks task [Milner, 1971] and requires maintenance of visually presented sequences of random locations in a 5×5 grid for further manual recall. To study compression in WM, sequences contained a specific pair of items forming a chunk. For each participant, there were three different chunks that appeared according to three frequency values (1/2, 1/4 and 1/8), the remaining 1/8 sequences being entirely randomized (no chunks).

Thirty participants (mean age = 22.7 years) took part in this experiment. The frequency values (1/2, 1/4 and 1/8) were manipulated within subjects and the experiment was decomposed in three successive phases: a span evaluation phase, a chunk learning phase and finally, the location task per se for a total duration of approximately 25 minutes.

All along the experiment, the participant was sited at about 60 cm in front of a computer screen in which the instructions as well as the experimental material were displayed using a JavaScript program. The 5×5 grid was centered on screen, each constitutive square was grey on a white background.

The first *span evaluation phase* aimed at measuring each individual's visuo-spatial WM raw capacity. To this end, each participant had to perform a classical location span task. Participants were presented with sequences of locations of ascending length started with sequences of three locations and, using the computer mouse, they had to manually recall the sequences in correct order by clicking sequentially on the appropriate squares immediately after the presentation of the last item of each sequence. There were three sequences of each length and a stop rule was applied according to which the span evaluation ended when the participant failed to recall the locations of all the three sequences at a particular level. Each correctly recalled series counted as one third; the total number of thirds added up to 2 (considering arbitrarily that the sequences of one and two locations that had not been performed were successfully recalled) provided a span score (e.g., [Barrouillet *et al*., 2011; Conlin, Gathercole, & Adams, 2005]). For example, the correct recall of all of the series of three locations, of two series of four locations, and of one series of five locations resulted in a span of $2 + (3 + 2 + 1) \times 1/3 = 4$. The presentation of the stimuli was temporally constrained: each location appeared for 500 ms and was followed by a 250 ms delay. After the last post-location delay of a given sequence, a mask (made of randomized pixels) was displayed for 1500 ms to minimize retinal persistence. Finally, the blank recall grid was displayed on screen up to the end of the participant's response (see Figure 3).

The second phase was devoted to the learning of the three chunks. Each chunk consisted in a sequential pair of two given locations. The pairs were randomly chosen by the computer for each participant according to the two following restrictive criteria: the four corner positions as well as the central position of the grid were never used and the two positions were never on the same line, column or diagonal. These criteria where applied to avoid individual differences in memory strategies. First of all, the participant, who was instructed to learn the associations of locations because they should be useful for the following part of the experiment, was presented with seven sequences of two locations following the three frequency values of exposure (1/2 for the first chunk

hereafter called AA, 1/4 for the second chunk hereafter called BB and 1/8 for the third chunk hereafter called CC). To control the effective learning, the first position of each chunk was then presented and the participant had to click on the appropriate second associated location. The chunk learning phase ended when the participant reached 100 % of success on the three chunks three times in a row. The temporal characteristics of the stimuli were similar to the *span evaluation* phase : locations were presented for 500 ms and followed by a 250 ms delay. A 2000 ms delay was inserted between the presentation of two successive to-be-learned chunks and there were no temporal restriction for recall.

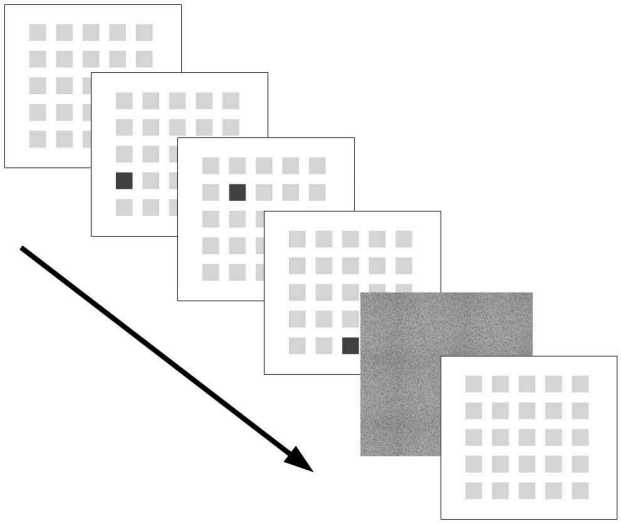

FIGURE 3. Sequence of images shown to the participant before recall, in the span evaluation phase for a sequence length of 3

Finally, the *location task* itself was administered to the participant who was presented with 64 sequences of constant length of locations. The number of locations that had to be memorized by each participant in each sequence was determined according to its personal raw visuo-spatial memory capacity evaluated in the first phase. Because we expected that the learning of chunks should enhanced memory performance, we choose to present sequences containing the span score of the given participant raised by two further locations. Eight blocks of eight to-be-memorized sequences were presented. In each block, four sequences started with the chunk AA, two sequences started with the chunk BB, one sequence with the chunk CC and the remaining sequence began by random locations that did not pertain to any learned chunks (hereafter called XX) and that were different across blocks. As previously, each location was presented for 500 ms and followed by a 250 ms delay. A 1500 ms mask followed the last post-location delay of each sequence to minimize retinal persistence and the recall phase was self-paced. The same spatial restrictions were applied to the random choice of the positions in a given sequence: the four corners and the central positions of the grid were avoided and two successive positions were never on the same line, column or diagonal.

Besides the span scores evaluating the raw individual visuo-spatial memory capacity in the first phase and revealing a mean memory performance of 4.01 locations across the thirty participants, three scores have been computed to measure memory performance in the location task: a *mean quantity score* evaluating the raw number of locations that can be memorized in correct order (this variable did not take the length of

the sequence into account), a percentage of locations recalled in correct order and a percentage of locations recalled irrespective of the order. While these different scores could have distinct theoretical justifications, we did not observe any significant differences on behavioural results across them. Indeed, on average, whatever the considered memory measure, behavioural results revealed that recall performance were better on sequences containing high-frequency chunks than on sequences containing low-frequency chunks. Indeed, while participants recall on average 71 % of locations in correct order in the AA chunk condition, their performance fell to 67 %, 67 % and 64 % for the BB, CC and XX conditions respectively. The AA condition showed significantly superior memory performance from all the three other conditions of frequency values of exposure ($t(29) = 2.97$, p-value < 0.01; $t(29) = 2.03$, p-value < 0.05 and $t(29) = 3.56$, p-value < 0.001, respectively). However, none of memory performance observed in these three conditions differed significantly from another (all p-values greater than 5 %).

These results show that all chunks do not seem to take the same space in working memory, since the encoding of a very frequent chunks like AA leads to an increase of recall performance. More items can be stored when the chunk AA is part of the stimuli probably because AA has been better compressed and therefore leaves more space for the other items.

4. SIMULATION

In order to go one step further in the study of the WM capacity, this section describes the simulation of the previous experiment, using a model of chunking based on the theoretical background presented previously. The idea is to simulate the creation of chunks by participants instead of assuming that chunks were perfectly learned by participants.

This model of chunking, called MDLChunker [Robinet *et al.,* 2011] describes the time course of chunk creation, stimulus after stimulus. In this model, each stimulus is a sequence of "letters" that could represent a word of an artificial language [Robinet & Lemaire, 2009], a set of visual items [Robinet *et al.*, 2011] or any component of an individual item. Here, a stimulus is a sequence of grid locations which are coded by a number between 1 and 25. All stimuli that have been presented to a given participant are therefore represented by a list of 64 sequences of about 5 to 7 numbers (depending on individual's visuo-spatial WM raw capacities measured by the *span evaluation phase)*. For instance (AA chunk is "22 5", BB chunk is "18 10" and CC chunk is "6 23"):

22 5 18 11 23 10
18 10 17 6 23 16
22 5 23 2 15 1
6 23 2 21 13 1
22 5 21 10 2 9
22 5 17 6 15 14
18 10 23 2 19 6
11 8 19 5 2 13
...

This is the material the model learns from. In the model, a chunk is a group of "letters" that tend to occur together in the stimuli. As mentioned previously, MDLChunker is based on rewriting the stimuli using two parts: chunks (model) and

data given the chunks (noise). For instance, the previous example could be rewritten in different ways. The first one would be not to consider chunks:

Chunks = {}
Data|chunks = { (22,5,18,11,23,10), (18,10,17,6,23,16), (22,5,23,2,15,1), … }

The second way would be to create a chunk for the sub-sequence 22,5 which seems to appear quite often. The codelength of the first part would be longer, but the second part would be a bit lower:

Chunks = {A = 22,5}
Data|chunks = { (A,18,11,23,10), (18,10,17,6,23,16), (A,23,2,15,1), … }

In MDLChunker, several chunks could be considered (even chunks containing chunks themselves, although this does not occur in this work). For instance:

Chunks = {A = 22,5 ; B = 18,10}
Data|chunks = { (A,18,11,23,10), (B,17,6,23,16), (A,23,2,15,1), … }

The best way is the one that has the shortest overall codelength. MDLChunker computes the lengths of the codes for representing the chunks (previously called p_{Σ}) and the lengths of the codes for representing the input data knowing these chunks ($p_{s|\Sigma}$), and minimizes their sum. Codelengths are estimated by means of Shannon's formula, saying that a symbol s, occurring with probability $P(s)$, can be ideally compressed with a binary code whose length is $C(s) = -\log_2(P(s))$. In our case, $P(s)$ is estimated by the frequency of s.

MDLChunker processes the sequences that a participant has been exposed to, constantly testing whether it is worth creating chunks. It looks for possible chunks at any position in sequences but since we only create regularities on the first two positions, it can only find chunks there. As soon as the creation of a chunk (a group of two grid locations) leads to a smaller overall codelength, the chunk is created. Figure 4 presents an example of the time course of chunk creation over the 64 sequences of a trial. In that simulation, no learning phase occurred as opposed to the human experiment. Chunk AA was created at iteration 11. At that time, it had been seen 5 times. Chunk BB was created at iteration 23 and chunk CC was created at iteration 56. On the 30 simulations, chunks AA was created between iterations 9 and 12, chunk BB was created between iterations 17 and 25 and chunk CC was created after iterations 40 (and may be not created at all in some cases).

FIGURE 4. Example of time course of chunk creation

Now that a model of chunking is available, we can supplement it with two models of working memory with different capacities and ask the model to recall items according to their capacities. The integrated models will then be compared to human data.

In the first model, capacity is a fixed number of chunks, *M*. For instance, if that value is 4, only 4 items would be memorized at iteration 7 of Figure 4. However, after chunk AA has been learned (at iteration 11 and after), the recall score would be 5 if the

chunk AA is part of the stimulus because AA would count only 1. Therefore one more item could be memorized.

In the second model, capacity is a fixed quantity of information of *N* bits. After each sequence is presented, the recall score is the maximum number of first items whose total codelength is equal or less than *N*. If a chunk exists, it is obviously considered. Codelengths change constantly because there are based on frequencies.

Each model is based on a parameter for the capacity (*M* or *N*). This parameter was learned for each model on data from 30 participants such that the recall score is the same as the participants' recall score after the first block of 8 sequences (0.67). We found $M = 3.85$ and $N = 12.7$ bits.

Figure 5 shows the percentage of correct recall for all blocks of 8 sequences for participants and models. It is worth noting that participants were exposed to the chunks prior to the experiment, which is not the case for the models. This is a drawback of the simulation, but it is not straightforward to mimic that prior learning. Therefore, we will mainly consider a comparison between the two models.

Both models show a burst of learning after the second sequence because there is much to learn. However, the model based on a fixed number of items keeps improving its performance. The model based on a quantity of information is rapidly as stable as participants.

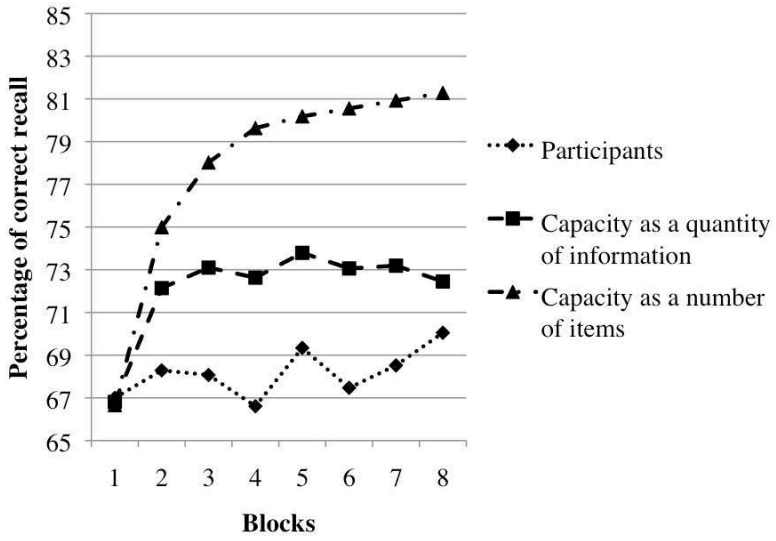

FIGURE 5. Percentage of correct recall for all blocks of 8 sequences for participants and models

In addition, Figure 6 (left side) shows that the model whose capacity is based on a fixed number of items converges to a situation with identical performance whatever the chunk presented in the sequence: the chunk with 50 % frequency is learned earlier, but when all three chunks have been learned, there is no difference at all. However, the model based on a fixed quantity of information (right side) always makes a difference between the three sequences: those containing the most frequent chunk are better recalled. Indeed, given the high frequency of the chunk, their codelengths are shorter.

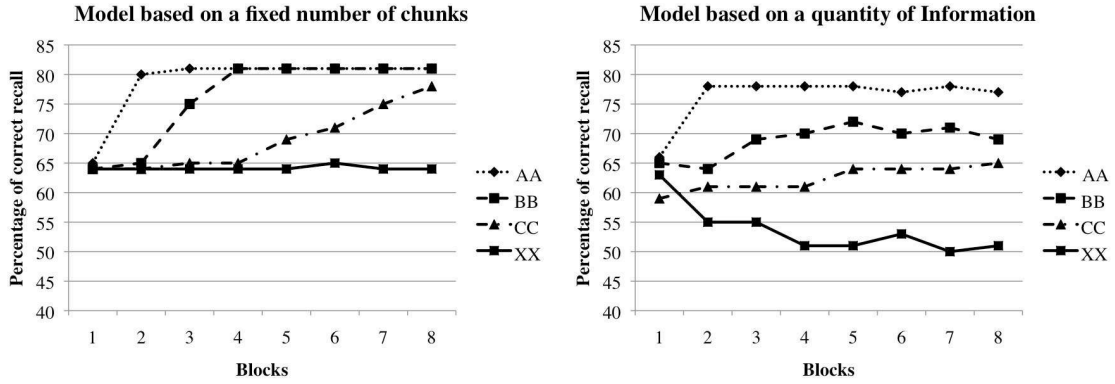

FIGURE 6. Percentage of correct recall for all blocks of 8 sequences, for each kind of stimuli.
Model based on a fixed number of chunks is on the left side.
Model based on a quantity of information is on the right side.

5. CONCLUSION

A theoretical framework, an experiment and a simulation lead us to consider that the capacity of the human working memory may be better expressed as a quantity of information rather than a fixed number of chunks. It is likely that humans are able to compress information in order to improve their performance of storage in memory and if so, an information theoretic measure is more likely to represent what is stored.

Many questions remain open. To what extent do humans rely on information compression? Are we optimal information compressors? More experiments should be conducted to compare human performance and mathematical models of information compression. As we mentioned previously, this is not an easy task because it has to be mainly based on novel material, for which the frequency can be easily controlled. It is true that another way could be to rely on existing corpora in order to estimate the frequency to which people are exposed to some stimuli, but the design of such experiments is probably not trivial.

Our model suggests a way of compressing information which is based on the frequency of chunks, defined as conjunctions of elementary units. However, other mechanisms of compression may occur that ought to be studied. In particular, chunks may be more complex than conjunctions.

This work is in line with a general point of view on cognition which is that humans tend to select simple structures [Chater & Vitanyi, 2003]: given several ways of understanding the world, we would retain the simplest one. Modeling this approach requires two distinct mechanisms: a generator of hypotheses and a way to select the simplest one. The nice thing is that information theory offers a way to quantify simplicity: simple explanations are those with the shortest codelengths.

BIBLIOGRAPHIE

ANDERSON J. R., BOWER G. H. (1972), "Recognition and retrieval processes in free recall", *Psychological Review* 79, pp. 97-123.

BADDELEY A.D. (1986), *Working memory*, Oxford, Calendron Press.

BARROUILLET P., PORTRAT S., CAMOS V. (2011), "On the law relating processing to storage in working memory", *Psychological Review* 118(2), pp. 175-192.

BRADY T.F., KONKLE T., ALVAREZ G.A. (2009), "Compression in visual working memory: using statistical regularities to form more efficient memory representations", *Journal of Experimental Psychology: General* 138(4), pp. 487-502.

CHAITIN G. (1966), "On the length of programs for computing finite binary sequences", *Journal of the ACM* 13(4), pp. 547-569.

CHATER N., VITANYI P.M. (2003), "Simplicity: A unifying principle in cognitive science?" *Trends in Cognitive Sciences* 7(1), pp. 19-22.

CONLIN J.A., GATHERCOLE S.E., ADAMS J.W. (2005), "Stimulus similarity decrements in children's working memory span", *Quarterly Journal of Experimental Psychology: Human Experimental Psychology* 58(A), pp. 1434-1446.

GOBET F., LANE P., CROKER S., CHENG P., JONES G., OLIVER I., PINE J.M. (2001), "Chunking mechanisms in human learning", *Trends in Cognitive Sciences* 5(6), pp. 236-243.

KOLMOGOROV A. (1968), "Three approaches to the quantitative definition of information", *International Journal of Computer Mathematics* 2(1), pp. 157-168.

KOLMOGOROV A. (1974), "Complexity of algorithms and objective definition of randomness", *Uspekhi Mat. Nauk* 29(4), pp. 155.

LEUNG-YAN-CHEONG S., COVER T. (1978), "Some equivalences between Shannon entropy and Kolmogorov complexity", *IEEE Transactions on Information Theory* 24(3), pp. 331-338.

LEVIN L. (1974), "Laws of information conservation (nongrowth) and aspects of the foundation of probability theory", *Problemy Peredachi Informatsii* 10(3), pp. 30-35.

MATHY F., FELDMAN J. [2012], "What's magic about magic numbers? Chunking and data compression in short-term memory", *Cognition*122, pp. 346-362.

MILLER G.A. (1956), "The magical number seven, plus or minus two: Some limits on our capacity to process information", *Psychological Review* 63(2), pp. 81-97.

MILNER B. (1971), "Interhemispheric differences in localization of psychological processes in man", *British Medical Bulletin* 27(3), pp. 272-277.

PERRUCHET P., VINTER A. (1998), "PARSER: A model for word segmentation", *Journal of Memory and Language* 39(2), pp. 246–263.

RISSANEN J. (1978), "Modeling by shortest data description", *Automatica* 14(5), pp. 465-471.

ROBINET V., LEMAIRE B. (2009), "MDLChunker: a MDL-based model of word segmentation", in *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, Amsterdam (Netherland), Cognitive Science Society, pp. 2866-2871.

ROBINET V., LEMAIRE B, GORDON M. (2011), "MDLChunker: a MDL-based Cognitive Model of Inductive Learning", *Cognitive Science* 35(7), pp. 1352-1389.

SAFFRAN J.R., NEWPORT E.L., ASLIN R.N. (1996), "Word segmentation: the role of distributional cues", *Journal of Memory and Language* 35(4), pp. 606-621.

SHANNON, C. (1948), "A mathematical theory of communication", *Bell System Tech. Journal* 27(3), pp. 379-423.

SOLOMONOFF, R. (1960), "A preliminary report on a general theory of inductive inference", Zator Co, Report V-131.

SOLOMONOFF, R. (1964), "A formal theory of inductive inference. Parts I and II", *Information and Control* 7(2), pp. 224-254.

SWINGLEY D. (2005), "Statistical clustering and the contents of the infant vocabulary", *Cognitive Psychology* 50(1), pp. 86-132.

TURING A. (1936), "On computable numbers: With an application to the Entscheidungsproblem", *Proceeding of the London Mathematical Society* 2, pp. 230-265.

VERESHCHAGIN N., VITANYI P. (2002), "Kolmogorov's structure functions with an application to the foundations of model selection", in *Proc. 47th IEEE symp. found. comput. Sci.* (FOCS02).

VITANYI P. (2005), "Algorithmic statistics and Kolmogorov's structure function", in *Advances in Minimum Description Length: Theory and Applications,* The MIT Press, pp. 151-174.